# Timer – Manual de Usuario

## Índice de contenido

# 1. Características

- Timer compatible con el microprocesador T80.
- Valor a decrementar configurable y automáticamente recargable cuando la cuenta llega a cero.
- Permite la lectura de la cuenta en curso. Selección de arranque automático o mediante trigger.
- Elección de flanco de subida o bajada de trigger.
- Configuración de 16 valores posibles de prescaler entre 2 y 65536 (2^16).
- Posee RESET por software.
- Posibilidad de manejo mediante polling, interrupciones en modo 1 o interrupciones en modo 2 utilizando un controlador de interrupciones.

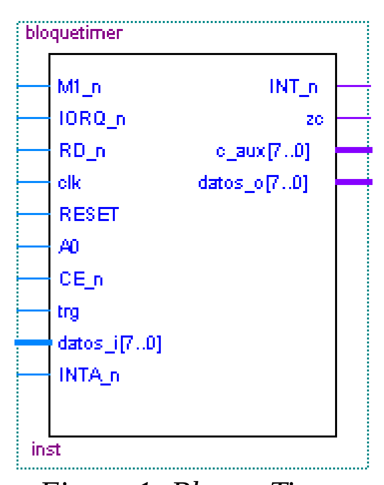



*Figura 1: Bloque Timer*

# 2. Descripción general

Este timer mide intervalos de tiempo contando períodos en la entrada clk de reloj del sistema y puede ser conectado con un microprocesador T80 de modo que este pueda configurarlo y utilizarlo como un periférico más.

La salida ZC da un pulso a '1' lógico cuando la cuenta expira permitiendo la conexión en cascada con un bloque counter.

El gran rango de su prescaler permite realizar medidas de tiempos del orden del reloj utilizado, así como de tiempos de hasta cuatro órdenes de magnitud mayor.

Se ofrece la posibilidad de contar tiempo luego de un evento mediante el arranque del proceso con un flanco de trigger inicial, el sistema puede configurarse para responder a flancos de subida o bajada en esta entrada.

En caso de reconfigurar alguna característica del sistema, las mismas tendrán validez luego de expirar la cuenta en curso. En caso de desearlo, puede realizarse un RESET por software cuando se realiza la reconfiguración y de este modo la actualización es instantánea.

El timer permite su utilización mediante polling chequeando el estado de la bandera ZC, cuando esta se encuentra en '1' lógico quiere decir que la cuenta expiró. La lectura del estado de esta bandera, borra la misma automáticamente sin necesidad de tener la constancia de hacerlo. El timer también permite su utilización mediante interrupciones, gracias a su salida Int_n la cual toma el valor '0' lógico cuando la cuenta expira (y se configura para habilitar las interrupciones) para luego borrarse automáticamente cuando se activa la entrada Inta_n.

# 3. Descripción funcional

Una vez conectado, el timer puede configurarse de modo de poder realizar medidas de tiempo del orden del reloj del sistema o de órdenes de hasta cuatro veces mayor. La medida puede iniciarse en forma automática o mediante un evento externo (pudiendo elegirse el tipo de flanco al cual responda el timer) y el valor de la medida puede ser consultada en cualquier momento. También se tiene la posibilidad de chequear mediante polling si se cumplió un tiempo a medir o que el timer produzca una interrupción en modo 1 cuando esto suceda. Tanto el tiempo a medir como el resto de las características mencionadas pueden modificarse en cualquier momento, teniendo validez sólo cuando la medición en curso culmine. De todos modos, puede forzarse la culminación de la medida actual en caso de desearlo.

# 4. Estructura interna

La estructura interna del timer puede descomponerse en cinco bloques: control, constante, counter, prescaler y control de interrupciones.

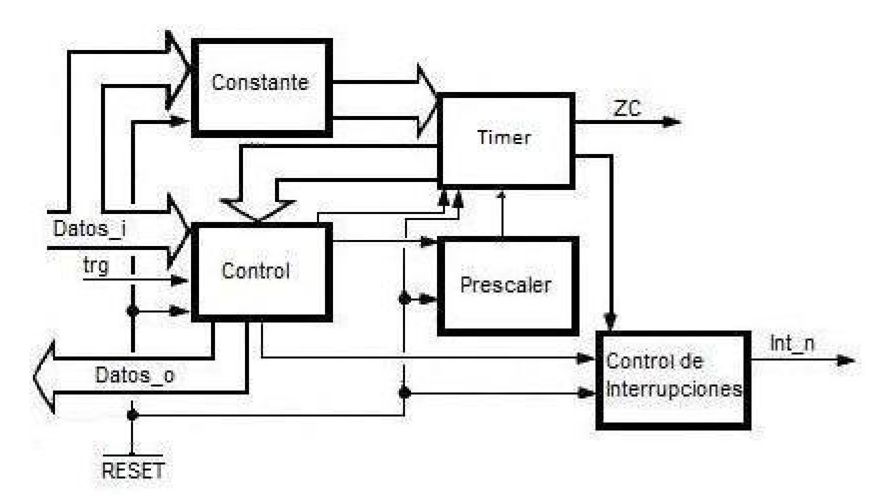



*Figura 2: Estructura interna del Timer*

## *4.1. Bloque Control*

Es el encargado de manejar los bloques Counter, Prescaler y Control de Interrupciones. Éste reconoce tanto los ciclos de escritura de la palabra de control (la cual es interna), de lectura de la cuenta en curso y reconocimiento de interrupción. Es responsable de setear el flag ZC, de forzar los resets por software, la elección del trigger, el arranque automático, la configuración del prescaler la habilitación y el reconocimiento de interrupciones. Este manejo lo logra mediante señales de control

con el resto de los bloques.

### 4.2. Bloque Constante

Es el bloque encargado de almacenar la constante a decrementar. Cuando se realiza un ciclo de escritura en la dirección adecuada, este bloque actualiza la constante.

### 4.3. Bloque Counter

Es el responsable de decrementar el número a contar el cual se recarga cada vez que la cuenta expira desde el bloque constante o cuando el bloque de control lo indica mediante señales de control entre estos.

### 4.4. Bloque Prescaler

Este bloque pone en su salida el reloj que utiliza el bloque counter, esta salida puede ser configurada para dividir la frecuencia de entrada por un valor configurable dentro de un amplio rango. Esta configuración se realiza mediante señales de entrada a este bloque, las cuales son manejadas por el bloque Control.

### 4.5. Bloque Control de Interrupciones

Este bloque maneja la señal de salida Int_n con la cual se realiza la solicitud de interrupción al procesador. Este recibe señales de control desde el bloque Control para manejar esta salida en caso de reconocimiento de interrupción o cuando las mismas están deshabilitadas.

## 5. Programación

El timer permite ser configurado mediante dos palabras diferentes, una palabra de control y el valor de la constante a decrementar. La escritura de estas palabras se realiza por parte del microprocesador mediante un ciclo “out” a la dirección correspondiente. Las direcciones en el espacio de entrada/salida de cada palabra requieren la decodificación por parte del usuario mediante el manejo de las entradas A0 y CE_n. La señal CE_n determina si el timer es el que debe recibir el ciclo, la señal A0 determina si se desea escribir la palabra de control o la constante a decrementar. También puede consultarse el estado de la cuenta y el flag de ZC en estas mismas direcciones mediante un ciclo de lectura.

| CE_n | A0 | Escritura | Lectura |
|---|---|---|---|
| **0** | **0** | **Constante** | **Flag ZC** |
| **0** | **1** | **Control** | **Cuenta** |

*Tabla 1: Decodificación de los elementos*

### 5.1. Constante a decrementar

Es el valor a decrementar por parte del timer, la misma se lee internamente cuando se reinicializa la cuenta. Esta palabra puede ser escrita en cualquier momento. (La constante a cargar

corresponde a la cantidad menos uno de cuentas a realizar, la misma debe ser distinta de cero).

## *5.2. Palabra de control*

Es la encargada de determinar el modo de funcionamiento del timer, se desglosa del siguiente modo:

### 5.2.1. Bit 7, habilitación de interrupciones

Con este bit seteado en '1' lógico, cada vez que la cuenta expire, la señal Int_n tomará el valor '0' lógico a modo de solicitud de interrupción por parte del timer, Int_n volverá a su valor de reposo ('1' lógico) cuando se active la entrada Inta_n. En caso contrario (bit 7 en '0' lógico) la señal Int_n permanecerá siempre en '1' lógico.

### 5.2.2. Bit 6, configuración de trigger

Con este bit seteado en '1' lógico, la señal de entrada trigger responderá a flancos de subida de la misma, en caso contrario responderá a flancos de bajada.

### 5.2.3. Bit 5, RESET por software

Con este bit seteado en '1' lógico, cada reescritura de la palabra de control forzará la reinicialización del timer, configurándose según el resto de los bits. En caso contrario (bit seteado en '0') los nuevos valores escritos en la palabra de control tendrán efecto recién en la próxima recarga del counter al llegar la cuenta a cero. La primera vez que se escribe la palabra de control la configuración tiene validez inmediatamente, independientemente del bit de RESET por software.

### 5.2.4. Bit 4, arranque por trigger

Con este bit seteado en '1' lógico, inicialmente el timer esperará un flanco activo en la entrada trigger (trg) para comenzar la cuenta, luego se reinicia automáticamente. En caso contrario (bit 4 en '0' lógico) la cuenta comenzará automáticamente o bien al escribirse la palabra de control si el bit 5 de RESET por software está en '1' o si es la primera vez que se escribe la palabra de control luego de un RESET hardware, o bien en la próxima recarga del timer si el timer ya estaba contando y el bit 5 de RESET por software está en '0'.

### 5.2.5. Bit [3..0], prescaler

El número decimal representado por estos cuatro bits es la potencia de dos por la cual se dividirá la frecuencia del reloj, en caso de ser cero (bit[3..0]='0000') la frecuencia se dividirá entre 2^16.

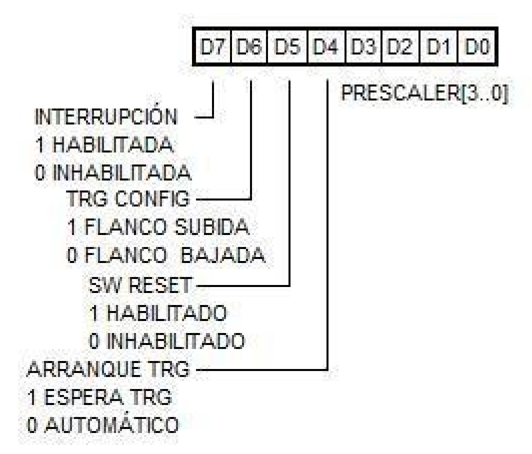



*Figura 3: Registro de control*

# 6. Descripción de pines e integración

## 6.1. Pines

- Clk (clock): reloj del sistema
- RESET: reset hardware del sistema
- CE_n (chip enable): activo por nivel bajo, cuando se encuentra activa reconoce los ciclos de lectura de cuenta y flag ZC, y escritura de constante y palabra de control.
- A0 (bit menos significativo del bus de direcciones): selecciona cual es el registro interno accedido en ciclos de lectura y escritura.
- Trg (trigger): señal para sincronizar el arranque mediante un evento de hardware.
- datos_i[7..0]: bus para recibir datos a ser escritos tanto en el registro constante como en el registro control.
- datos_o[7..0]: bus para mostrar la cuenta en curso o el flag ZC del timer.
- M1_n: (Machine Cycle One): activo por nivel bajo, utilizado para diferenciar ciclos de escritura en I/O de ciclos de reconocimiento de interrupción del T80. Cuando el T80 realiza un ciclo de reconocimiento de interrupción las señales M1_n y IORQ_n toman el valor lógico 0. Ya que el timer no cuenta con la señal WR_n, si no fuese por M1_n, éste no podría diferenciar entre ciclos de escritura y ciclos de reconocimientos de interrupción del T80.
- IORQ_n (Input/Output Request): activo por nivel bajo, utilizado para los ciclos de lectura y escritura (junto con CE_n, A0 y RD_n).
- RD_n (Read): activo por nivel bajo, utilizado para realizar ciclos de lectura y escritura al timer (junto con IORQ_n, CE_n y A0).
- ZC (Zero Count): activo por nivel alto, produce un pulso en '1' cuando la cuenta expira.
- Int_n (Interrupt Request): activa por nivel bajo, responsable de la solicitud de interrupción cuando se encuentra habilitada la misma.

- Inta_n (interrupt acknowledge): activa por nivel bajo, indica al timer que la interrupción fue atendida.

### 6.2. Integración a un sistema con microprocesador T80

El timer está diseñado para ser compatible con el microprocesador T80 por lo cual la integración a un sistema con él es simple y se describe a continuación.

- Las señales de control M1_n, IORQ_n y RD_n se conectan directamente a las señales de igual nombre del T80.
- En caso de no usar controlador de interrupciones, la señal Int_n también se conecta a la del mismo nombre en el microprocesador.
- El manejo de Inta_n depende del modo de interrupciones usado en el T80. En modo 1 puede generarse a partir de IORQ_n y M1_n o con un pulso de selección de dispositivo I/O. En modo 2 esta señal debe ser manejada por el controlador de interrupciones.
- A la entrada datos_i[7..0] se conecta la salida datos_o[7..0] del procesador T80. La salida datos_o[7..] debe hacerse llegar a la entrada datos_i[7..0] del T80 habilitándola en forma adecuada con las señales de decodificación.
- Tanto clk como RESET son las correspondientes al sistema completo.
- A0 y CE_n corresponden a la decodicación en el espacio de entrada/salida que se desee.
- No hace falta conectar el pin ZC ya que la bandera ZC puede leerse mediante polling en la dirección adecuada y en todo momento.

## 7. Timing

### 7.1. Ciclos de Lectura

En la figura 4 se muestra el ciclo de lectura del contenido de la cuenta, en la misma se observan las señales de control CE_n, RD_n e IORQ_n en nivel bajo y M1_n en nivel alto.

### 7.2. Ciclos de Escritura

En la figura 5 se muestra un ciclo de escritura tanto de una palabra de control como de una constante, en la misma se observan las señales de control CE_n e IORQ_n en nivel bajo y RD_n junto con M1_n en nivel alto.

### 7.3. Comienzo de la cuenta

En la figura 6 se muestra el inicio de una cuenta por trigger, la cual detecta el nivel alto de la entrada trg en un flanco y en el siguiente flanco comienza la cuenta. La señal c_aux muestra el valor interno de la cuenta, en el ejemplo el prescaler está dividiendo entre 2 y la cuenta cambia cada dos períodos del reloj del sistema.

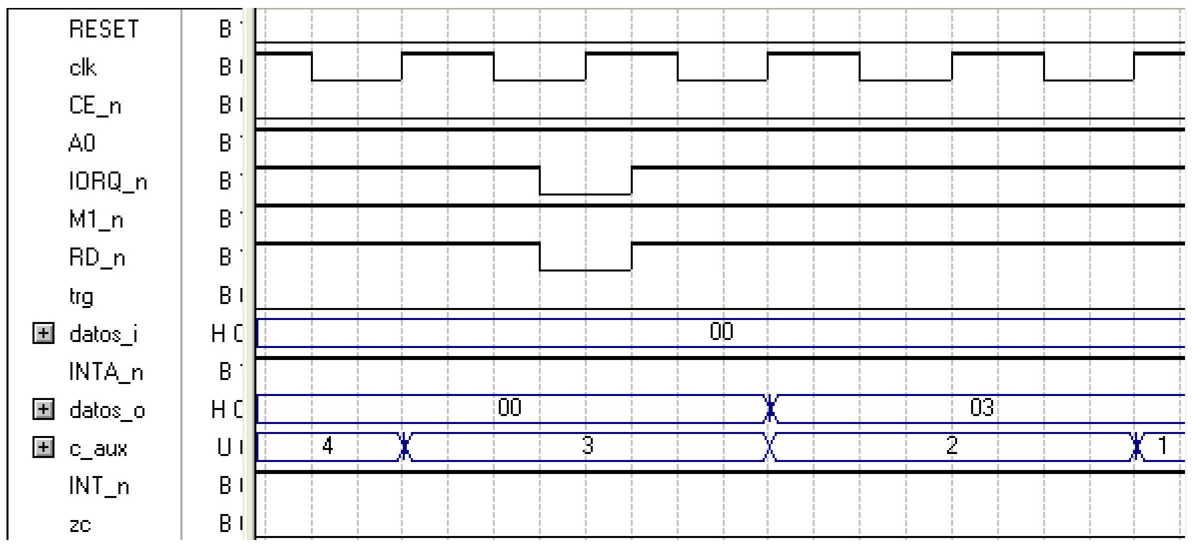



*Figura 4: Ciclo de lectura del estado de la cuenta, la solicitud se realiza con IORQ_n, RD_n en nivel bajo y M1_n en nivel alto. En el flanco siguiente se detectan estos valores de las señales y se escribe en datos_o el valor de la cuenta en curso.*

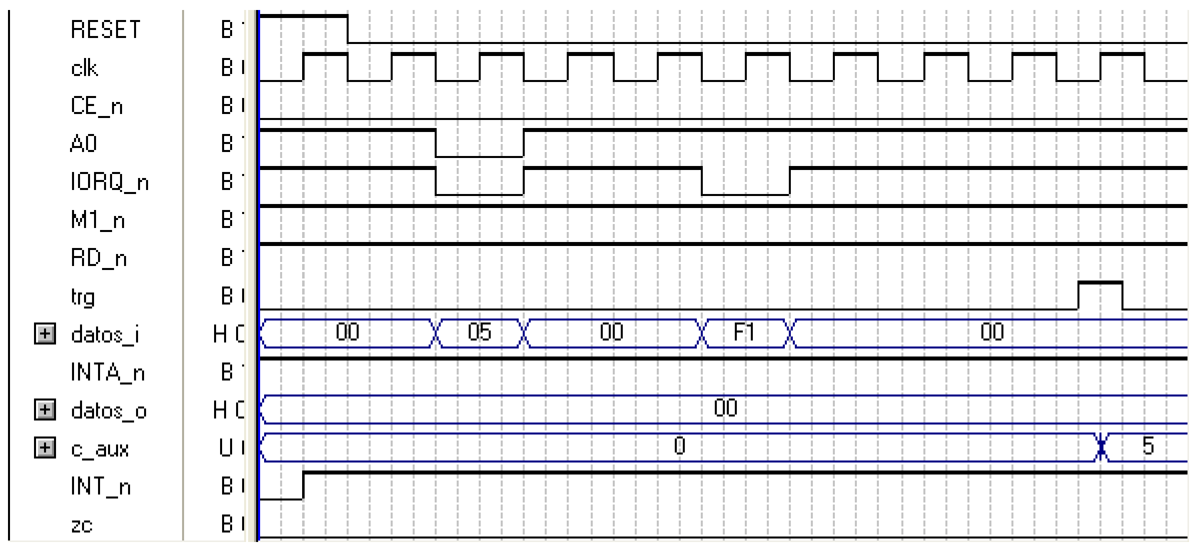



*Figura 5: Ciclo de escritura de la palabra de control, en la misma la señal IORQ_n se encuentra en nivel bajo mientras que las señales RD_n y M1_n se encuentran en nivel alto. En el flanco de subida del reloj donde ocurre esta combinación de señales se escribe la palabra.*

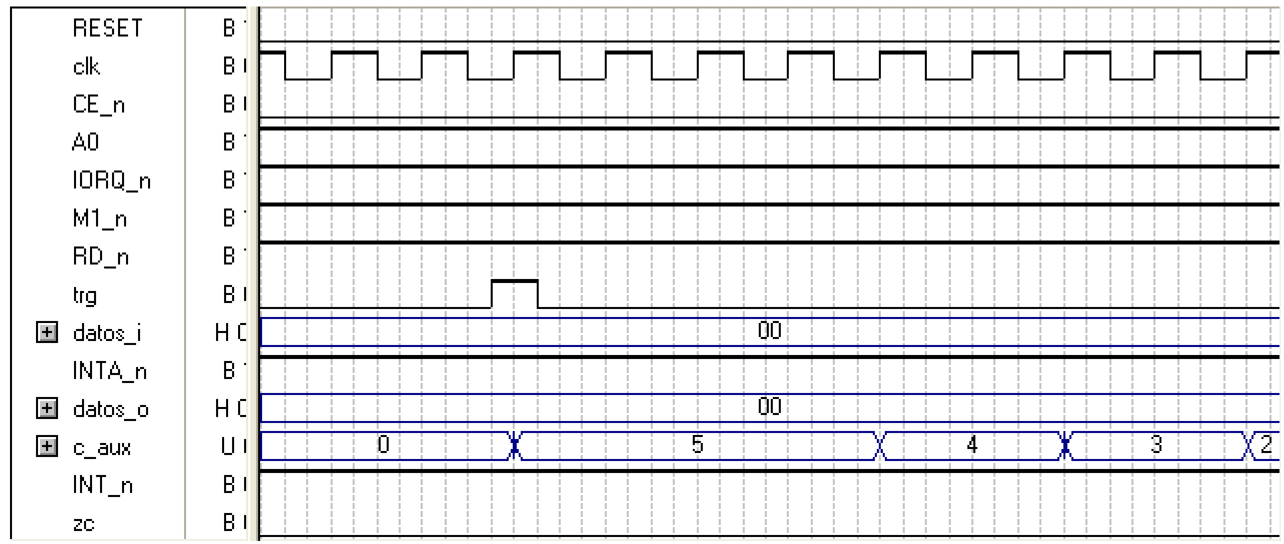



*Figura 6: inicio del conteo mediante un pulso de la entrada trg, en la misma se observa la espera del timer por la detección de un flanco de subida de esta entrada, chequeando su valor en cada flanco de subida del reloj del sistema. Una vez detectado, la constante es cargada y al siguiente flanco de reloj inicia la cuenta.*

# 8. Funcionamiento durante Interrupciones

## *8.1. Solicitud de interrupción*

En la figura 7 se muestra una solicitud de interrupción del timer cuando la cuenta expira, la señal Int_n toma el valor '0' lógico después del flanco de reloj.

## *8.2. Reconocimiento de interrupción*

En la figura 8 se muestra un ciclo de reconocimiento de la interrupción del timer mediante la señal Inta_n en nivel bajo, luego de esto la señal Int_n toma el valor '1' lógico después del flanco de reloj. Notar que en este caso Inta_n se activa en un ciclo de reconocimiento de interrupción del T80 por lo que IORQ_n y M1_n también se encuentran en nivel bajo,

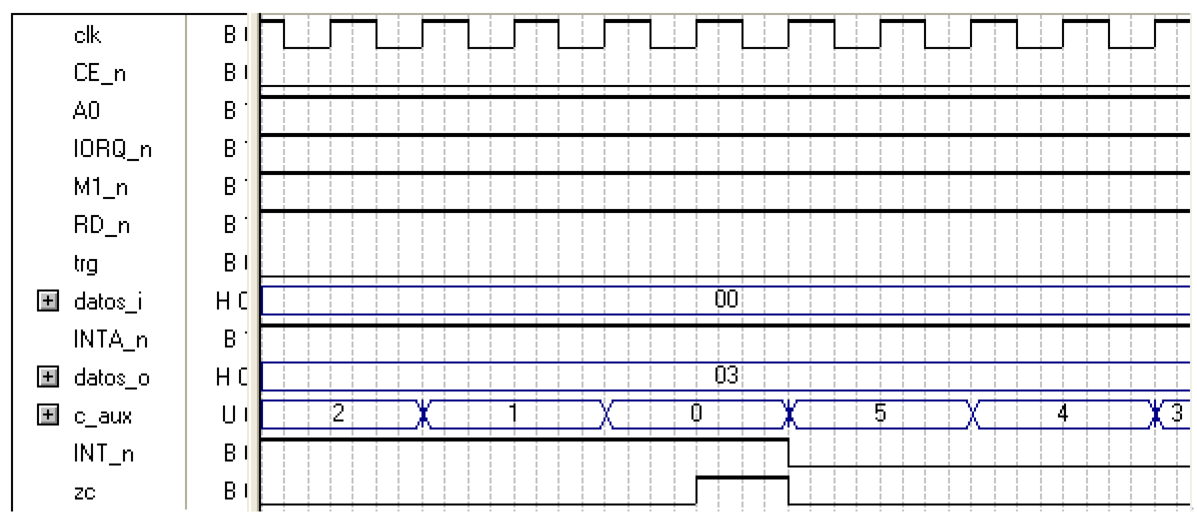



*Figura 7: Solicitud de interrupción por parte del timer: En cada flanco de reloj se chequea si la cuenta ha llegado a cero. Una vez que esto ocurre, la señal Int_n toma el valor lógico '0' si las interrupciones están habilitadas (se observa también un pulso de un ciclo de reloj de la señal ZC en el valor '1' lógico).*

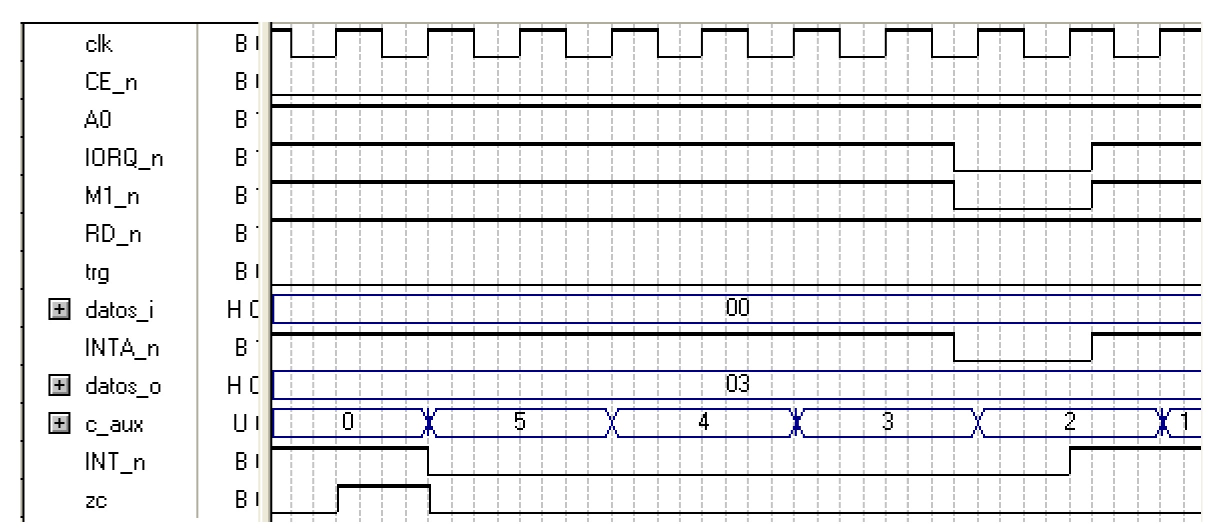



*Figura 8: Reconocimiento de interrupción por parte del timer: la señal Inta_n toma el valor lógico '0'. La detección se realiza en el flanco de subida del reloj del sistema y cuando ocurre la señal Int_n vuelve a su valor de reposo.(Se observa también que la cuenta sigue decrementándose).*

# Revisiones

| Fecha | Versión | Comentarios |
|---|---|---|
| 11/07/10 | 0 | Santiango Martínez, Federico Bliman y Matías Bakalián<br>Proyecto de curso Diseño Lógico 2010 |
| 20/05/11 | 1 | Andrés Touya<br>Cambio de CS por A0 y corrección sobre funcionamiento de disparo por trigger |
| 28/05/12 | 2 | Andrés Touya<br>Se agrega Inta_n como señal de entrada |